Attorney Docket No.: 42390.P12324 Patent Application

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

TITLE OF THE INVENTION

MULTI-TOKEN TOTALLY ORDERED GROUP COMMUNICATION PROTOCOL

INVENTORS

SAM MAZZA

Prepared by

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1026
(303) 740-1980

EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number: EL 886506969 US
Date of Deposit: September 26, 2001
I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Commissioner of Patents and Trademarks, Washington, D. C. 20231

Fran C. Rolfsen
(Typed or printed name of person mailing paper or fee)

Fran C. Rolfsen
(Signature of person mailing paper or fee)

9-26-01
(Date signed)

MULTI-TOKEN TOTALLY ORDERED GROUP COMMUNICATION PROTOCOL

COPYRIGHT NOTICE

FIELD OF THE INVENTION

[0002] The invention relates generally to communication protocols. More particularly, the invention relates to ordered group communication protocols that utilize logical tokens for communication.

BACKGROUND OF THE INVENTION

[0003] A necessary feature for communication protocols used in fault tolerant systems is a mechanism to guarantee that all messages can be logically ordered in a linear sequence. Fault tolerant systems use entity-replication to provide high availability services. Each member of a replication group must maintain a consistent state. This allows a replica to arrive at the last known valid state of the primary entity in the event a fault occurs. One mechanism that has been developed to guarantee that the replicas' state is consistent with that of the primary entity is to impose a total order of messages.

[0004] An example of a current technique used to impose a total order of messages will now be described with reference to Figure 1A. Figure 1A illustrates a logical token ring 105 superimposed on a Local Area Network (LAN) 100, such as an Ethernet. A token 110 is circulated around the nodes 120-127 on the LAN 100. A node that wishes to send a message may only do so when it has ownership of the token 110. Upon receipt of the token 110, the sender generates the next sequence number for the message about to be

sent. This procedure guarantees that all messages can be logically ordered in a linear sequence, thus providing for a total order of messages for all nodes on the LAN 100.

[0005] Each of the nodes 120-127 could be a member of one or more groups. Figure 1B illustrates one example of group membership. Groups 130, 140, or 150 could be a replication group or another type of distributed application group that requires total ordering of messages. In this example, node 124 is a member of group 130. Node 121 is a member of groups 130 and 140. Nodes 120 and 127 are members of group 140. Nodes 122, 123, and 125 are members of group 150.

[0006] The total ordering of messages only needs to be imposed on messages within each group 130, 140, and 150. However, the prior art approach imposes a total ordering of messages on the LAN 100 as a whole without regard for groups. Consequently, if node 122 wishes to send a message to the other members of its group, it may have to wait for the token to travel around all of the nodes 123-121 before it can receive the token 110 and transmit its message. An additional shortcoming is that members of different groups may not transmit messages to their respective groups concurrently. For example, a member of group 130 may not transmit a message to its group simultaneous with node 122's message.

[0007] A local area network may have many groups communicating. The use of a single token when multiple groups are present unnecessarily serializes unrelated communication. This adds unneeded latency to group communication.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

[0008] The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

[0009] **Figure 1A** is a block diagram that illustrates a logical token ring superimposed on a local area network.

[0010] **Figure 1B** is a block diagram that illustrates group membership.

[0011] **Figure 2** is an example of a typical computer system upon which one embodiment of the present invention can be implemented.

[0012] **Figure 3** is a block diagram illustrating multiple logical token rings superimposed on a local area network to facilitate concurrent transmissions among multiple groups according to one embodiment of the present invention.

[0013] **Figure 4** is a flow diagram that illustrates message transmission processing according to one embodiment of the present invention.

[0014] **Figure 5A** illustrates two replication groups where the primaries are both members of the logical token rings associated with each replication group.

[0015] **Figure 5B** is a flow diagram that illustrates sending a message to the replication group from one of the primaries according to one embodiment of the present invention.

[0016] **Figure 6A** illustrates two replication groups where a replica replicates both primaries.

[0017] **Figure 6B** is a flow diagram that illustrates, according to one embodiment of the present invention, receiving a message at the replica that is replicating both primaries

[0018] **Figure 7** is a flow diagram that illustrates updating an all received up-to (aru) field according to one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0019] A method and apparatus are described for performing group communication. According to one embodiment of the present invention, multiple logical token rings are imposed on a LAN (one for each group on that LAN). A token representing permission to broadcast a message circulates among the members of the group. The multiple tokens enable communications for each group to be independently serialized from the other groups. This allows multiple groups to communicate simultaneously, thus reducing latency.

[0020] In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some of these specific details. In other instances, well-known structures and devices are shown in block diagram form.

[0021] The present invention includes various steps, which will be described below. The steps of the present invention may be performed by hardware components or may be embodied in machine-executable instructions, which may be used to cause a general-purpose or special-purpose processor or logic circuits programmed with the instructions to perform the steps. Alternatively, the steps may be performed by a combination of hardware and software.

[0022] The present invention may be provided as a computer program product which may include a machine-readable medium having stored thereon instructions which may be used to program a computer (or other electronic devices) to perform a process according to the present invention. The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, CD-ROMs, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, flash memory, or other type of media / machine-readable medium suitable for storing electronic instructions.

Moreover, the present invention may also be downloaded as a computer program product, wherein the program may be transferred from a remote computer to a requesting computer by way of data signals embodied in a carrier wave or other propagation medium via a communication link (e.g., a modem or network connection).

[0023] Importantly, while embodiments of the present invention will be described with reference to the Totem Protocol as described in *"Totem: A Reliable Ordered Delivery Protocol for Interconnected Local-Area Networks,"* Deborah A. Agarwal (Doctoral Dissertation, Department of Electrical and Computer-Engineering, University of California, Santa Barbara, August 1994), the method and apparatus described herein are equally applicable to other types of group communication protocols that utilize logical token rings to serialize communication or future enhancements to the Totem protocol.

Terminology

[0024] Before describing an exemplary environment in which various embodiments of the present invention may be implemented, some terms that will be used throughout this application will briefly be defined.

[0025] A node generally refers to a processing element on a network. For example, a node may represent one or more processors executing one or more processes each of which may belong to one or more groups.

[0026] The term logical token ring refers to those nodes that receive a particular token. The token is passed from one node to another in an ordering that can be regarded as a "logical ring." In the examples discussed in this application, the token represents permission to send a message. Other uses for the token may be possible.

[0027] "Passing" a token broadly refers to releasing control or ownership of the token. According to one embodiment, token passing may be performed by the current owner of the token transmitting it to the next node on a list. Other methods to share the token between group members may also be employed.

[0028] A group is a set of entities that maintain serialized communication across a network. An entity may be a node, a computer, a processor, a process, a software object, or another type of hardware device. Exemplary groups include replication groups, process groups, or nodes participating in the execution of a distributed application.

[0029] Total order means that all messages can be logically ordered in a linear sequence.

[0030] Agreed delivery means that when a message is delivered, the group member has delivered all messages within the group with an earlier timestamp.

[0031] Safe delivery means that before delivering a message, a group member knows that the other group members have received the message.

[0032] The term replication group refers to a group of entities, comprising a primary entity and one or more replica entities that replicate the state of the primary entity. **[0033]** A "hot replica" is a replica that is executing along with the primary.

[0034] A "warm replica" refers to a replica that is ready to run, but will need to retrieve the prior state and/or may retrieve and execute messages from the point in time of the prior state to the point of failure of the primary.

[0035] A "cold replica" refers to a replica that is not running. The replica will need to be launched in the case of failure of the primary. The cold replica will also need to retrieve the prior state and/or may retrieve and execute messages from the point in time of the prior state to the point of failure of the primary.

[0036] Totem is a fault tolerant multicast communication protocol that uses a single token on a LAN to serialize communications on the network. The senders are allowed to send a message only when they have ownership of the token. Totem provides for agreed delivery and safe delivery. The fields of a totem token include a type field, a ring identifier field, a token sequence number, a high-water mark indicating the largest sequence number of any message that has been broadcast on the ring, an all received up-to field that is used to determine which messages processors on the ring have received, a

field identifying the processor that set the all received up-to field, and a retransmission request list. Each message in Totem includes a sender identifier, a ring identifier, a message sequence number, and the contents of the message. The processors each maintain local variables including a sequence number that indicates the processor has received all messages with sequence numbers less than or equal to the sequence number, the value of the token sequence number when the processor last forwarded the token, and the sequence number of the message the processor most recently delivered.

An Exemplary Node

[0037] A computer system 200 representing an exemplary node 120-127 in which features of the present invention may be implemented will now be described with reference to Figure 2. Computer system 200 comprises a bus or other communication means 201 for communicating information, and a processing means such as a processor 202 coupled with bus 201 for processing information. Computer system 200 further comprises a random access memory (RAM) or other dynamic storage device 204 (referred to as main memory), coupled to bus 201 for storing information and instructions to be executed by processor 202. Main memory 204 also may be used for storing temporary variables or other intermediate information during execution of instructions by processor 202. Computer system 200 also comprises a read only memory (ROM) and/or other static storage device 206 coupled to bus 201 for storing static information and instructions for processor 202.

[0038] A data storage device 207 such as a magnetic disk or optical disc and its corresponding drive may also be coupled to computer system 200 for storing information and instructions. Computer system 200 can also be coupled via bus 201 to a display device 221, such as a cathode ray tube (CRT) or Liquid Crystal Display (LCD), for displaying information to a computer user. Typically, an alphanumeric input device 222, including alphanumeric and other keys, may be coupled to bus 201 for communicating information and/or command selections to processor 202. Another type of user input device is cursor

control 223, such as a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 202 and for controlling cursor movement on display 221.

[0039] A communication device 225 is also coupled to bus 201 for access to the network, such as LAN 100. The communication device 225 may include a modem, a network interface card, or other well-known interface devices, such as those used for coupling to an Ethernet, token ring, or other types of networks. In any event, in this manner, the computer system 200 may be coupled to a number of clients and/or servers via a conventional network infrastructure, such as a company's Intranet and/or the Internet, for example.

[0040] It should be appreciated that this invention is not limited to the exemplary node described in this example. In alternative embodiments, nodes may also comprise various combinations of computers, processors, other hardware devices, software processes, or other software objects. Nodes may also be coupled to alternate network infrastructures, such as a wireless network.

Multiple Logical Token Rings on a LAN

[0041] Figure 3 illustrates multiple logical token rings 300, 310, and 320 on a LAN 100, according to one embodiment of the invention. In this example, there is a logical token ring 300, 310, and 320 for each group 130, 140, and 150. Token 305 is circulated among the nodes 121 and 124 of group 130. Token 315 is circulated among the nodes 120, 121, and 127 of group 140. Token 320 is circulated among the nodes 121, 123, and 125 of group 150. Each token 305, 315, and 325 is used to serialize messages among members of the corresponding group.

[0042] It should be appreciated that the present invention is not limited to a token circulating among the nodes on a LAN. For example, the nodes of a group may span multiple LANs. The token may also be passed among members of a group on a communication means other than a LAN, such as a bus or a data communication system

within a single processor or multiprocessor computer system. Additionally, the logical token ring may be configured so that the token arrives at a node more than once before it arrives to another node at all. Finally, it is contemplated that tokens might be implemented as shared resources with an arbitration mechanism to resolve conflicting requests for the tokens.

Message Transmission Processing

[0043] A node that wishes to send a message to a group may only do so when it has ownership of that group's token. Message transmission processing according to one embodiment of the invention will now be illustrated with reference to Figure 4. At block 410, the node receives a token.

[0044] At block 420, the node checks to see if it has any messages at the head of a message queue that are destined for the received token's group. The node may be managing one or more message queues. If there is a message at the head of a queue destined for the received token's group, processing continues with block 430. Otherwise, processing continues with block 450. This example assumes that the node is using FIFO (First In First Out) to send its messages. In another embodiment of the invention, the node may be using another approach to managing its messages. In any event, upon receiving a token, the node checks its message queues according to the message management approach utilized and makes a determination regarding which messages, if any, are ready for transmission.

[0045] At block 430, the node increments a sequence number associated with the token. The sequence number is used to provide agreed delivery and a total order of messages among members of a process group.

[0046] At block 440, the node sends the message using the sequence number associated with the token. The message may be a unicast, multicast, or broadcast message. In block 450, the node passes the token to the next member of the group. In another embodiment, before passing the token to the next member of the group, the node may return to block

420 and continue sending messages until there are no messages at the head of its queue destined for the token's group, or until a specific event, such as a time out, has occurred.

[0047] Although the previous example illustrates the node incrementing a sequence number associated with the token and then sending a message using the new sequence number (pre-increment), it should be appreciated that alternate approaches may also be used. Another algorithm may be used to generate the sequence number. For example, the node may also send a message using the current sequence number and then generate a new sequence number before sending another message or passing the token to the next member of the group (post-increment).

Replication Groups

[0048] Fault tolerant systems use entity replication to provide high availability services. The replica entities maintain a consistent state with the entity being replicated (primary). Total order of messages guarantees that the replicas' state is consistent with that of the primary. A total order of messages is provided for each replication group. Messages are processed in the same order on the primary and its replicas. Typically, the message sequence number is the imposed order. In one embodiment, replication groups utilize a token to provide message sequence numbers. Other methods to impose total order are also possible. For example, a pre-selected entity may define an order on the messages and notify or forward the messages or the now specified order of the messages to the members of the group.

[0049] Figure 5A illustrates two replication groups 500 and 510. Group 500 contains a replica 521 for primary 520. Group 510 contains replica processors 531 and 532 for primary 530. Token 505 is circulated among the members of group 500. Token 515 circulates among the members of group 510.

[0050] In this example, the primaries 520 and 530 need to communicate with each other. For example, one primary could be a client and the other could be a server. According to this embodiment, both primaries are responsible for maintaining message synchronization among the groups 500 and 510. Therefore, primaries 520 and 530 are each members of both groups 500 and 510.

[0051] Figure 5B illustrates message transmission processing from primary 520. At block 540, the primary 520 receives token 505. At block 550, the primary 520 checks to see if it has any messages that need to be sent to group 500. If a message is destined for the primary's own replication group 500, the message does not need to be synchronized with group 510. Therefore, the primary may proceed with block 580.

[0052] If there are no messages that need to be sent to group 500, the primary 520 checks to see if it has any messages that need to be sent to group 510. If it does not have any messages for group 510, the primary 520 has no messages that require the use of token 505. Thus, processing continues with block 555 where the token 505 is passed to the next member of group 500.

[0053] If the primary 520 has any messages destined for group 510, processing continues at block 560. The message to group 510 needs to be synchronized between both the primary's own replication group 500 and group 510. Primary 520 needs token 515 to send a message to group 510. Token 505 is needed to ensure replica 521 receives notification of the message primary 520 sent to group 510 and thus maintains a consistent state with primary 520. Therefore, at block 560, the primary 520 must wait for token 515. At block 570, the primary 520 receives token 515. Message synchronization between the groups 500 and 510 can now be maintained. At block 580, the message is sent.

[0054] In this illustration, both primaries were responsible for maintaining message synchronization among the replication groups 500 and 510. In another embodiment of the invention, this responsibility could be delegated to one of the primaries. In that case,

only the primary responsible for maintaining the message synchronization would be a member of both groups 500 and 510.

[0055] In another embodiment of the invention, a replica may replicate more than one primary. This is illustrated in Figure 6A. Primary 620, replicas 625 and 630 and client 635 are members of group 600. Primary 640 and replicas 630 and 645 are members of group 610. Token 605 circulates among the members of group 600. Token 615 circulates among the members of group 610.

[0056] Storage areas 650 and 655 are provided to allow replica 630 to maintain separate ordered lists of messages from groups 600 and 610. Storage areas 650 and 655 may comprise a file, a part of a database, magnetic tape, magnetic disk, memory resident data structures, or another type of storage mechanism. Since replica 630 replicates both primary 620 and primary 640, it cannot act as a hot replica for both. Therefore, in this example, replica 630 is assumed to be a warm replica for both primaries 620 and 640. Importantly, in this example, message synchronization at replica 630 is by way of separate storage areas rather than tokens.

[0057] In this embodiment, replica 630 replicates both primary 620 and primary 640. Message receipt at replica 630 is illustrated in Figure 6B. At block 650, the replica 630 receives a message. At block 665, replica 630 determines if the message was for group 600. If it was, processing continues with block 660. Otherwise, processing continues with block 665.

[0058] At block 660, the message is stored in the group 600 storage area. If the primary 620 fails, the replica 630 can then execute all of the messages in the group 600 storage area to achieve the last known state of the primary 620. After the message is stored, processing ends at block 675.

[0059] At block 665, the replica 630 determines if the message it received was for group 610. If not, processing ends at block 675. If the message was destined for group 610, processing continues with block 670.

[0060] At block 670, the replica stores the message in the group 610 storage area. If the primary 640 fails, the replica 630 can then execute all of the messages in the group 610 storage area to achieve the last known state of the primary 640. After the message is stored, processing ends at block 675.

[0061] In this example, replica 630 did not maintain its state with either of the primaries 620 or 640. In alternate embodiments, it should be appreciated that replica 630 may maintain its state, or act as a hot replica, for one of the primaries 620 or 640 and act as a warm or cold replica for the other primary 620 or 640. After receipt of a message, replica 630 would then alter its state for the primary for which it was a hot replica and store the message for the primary for which it was a warm or cold replica. It should also be appreciated that the primary may push a state onto a replica periodically or a replica may periodically ask the primary for its state.

Safe Delivery

[0062] In one embodiment of the invention, a token may have an associated all received up-to (aru) field. The aru field may be used to provide safe delivery by informing all nodes on the ring of the last message that other group members have received. In this embodiment, in addition to the aru field associated with a token, each node also maintains a local aru variable for each group in which it participates that contains the last message the node received. Figure 7 illustrates a node updating the aru field.

[0063] At block 710, the node receives a token. At block 720, the node obtains the value for the aru field associated with the token. At block 730, the node determines if its local aru variable for the token's group is less than or equal to the aru field associated with the token. If it is, processing continues with block 740. Otherwise, processing continues with block 750.

[0064] At block 740, the node sets the aru field associated with the token equal to its aru variable for this token. Processing then ends at block 770.

[0065] At block 750, the node determines if it is sending a message to this token's group. If it is, in block 760, the aru field associated with the token and the node's local aru variable are set to equal the sequence number used for the message. Processing then ends at block 750. If the node is not sending a message to the token's group, the aru field is not updated and processing ends at block 770.

[0066] Previous illustrations have shown a token having a sequence number and/or aru field associated with it. In alternate embodiments, the token may have other fields associated with it. For example, in one embodiment, the token may be a Totem token having a type field, a ring identifier field, a token sequence number, a high water mark indicating the last sequence number of any message that has been broadcast on the ring, an aru field used to determine which messages processors on the ring have received, a field identifying the processor that set the aru field, and a retransmission request list.

[0067] In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.
